UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
10/530,375	04/06/2005	Carlos Antonio Alba Pinto	NL 020979	4877

24737 7590 02/12/2009
PHILIPS INTELLECTUAL PROPERTY & STANDARDS
P.O. BOX 3001
BRIARCLIFF MANOR, NY 10510

EXAMINER	
OTTO, ALAN	

ART UNIT	PAPER NUMBER
2187	

MAIL DATE	DELIVERY MODE
02/12/2009	PAPER

Please find below and/or attached an Office communication concerning this application or proceeding.

The time period for reply, if any, is set in the attached communication.

Office Action Summary	Application No.	Applicant(s)	
	10/530,375	ALBA PINTO ET AL.	
	Examiner	Art Unit	
	ALAN M. OTTO	2187	

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --

Period for Reply

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE *3* MONTH(S) OR THIRTY (30) DAYS, WHICHEVER IS LONGER, FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133). Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

Status

1)☒ Responsive to communication(s) filed on *08 December 2008*.

2a)☒ This action is **FINAL**. 2b)☐ This action is non-final.

3)☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

Disposition of Claims

4)☒ Claim(s) *1-20* is/are pending in the application.

4a) Of the above claim(s) _____ is/are withdrawn from consideration.

5)☐ Claim(s) _____ is/are allowed.

6)☒ Claim(s) *1-20* is/are rejected.

7)☐ Claim(s) _____ is/are objected to.

8)☐ Claim(s) _____ are subject to restriction and/or election requirement.

Application Papers

9)☐ The specification is objected to by the Examiner.

10)☐ The drawing(s) filed on _____ is/are: a)☐ accepted or b)☐ objected to by the Examiner.

Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).

Replacement drawing sheet(s) including the correction is required if the drawing(s) is objected to. See 37 CFR 1.121(d).

11)☐ The oath or declaration is objected to by the Examiner. Note the attached Office Action or form PTO-152.

Priority under 35 U.S.C. § 119

12)☒ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).

a)☒ All b)☐ Some * c)☐ None of:

1.☐ Certified copies of the priority documents have been received.

2.☐ Certified copies of the priority documents have been received in Application No. _____.

3.☒ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

* See the attached detailed Office action for a list of the certified copies not received.

Attachment(s)

1) ☐ Notice of References Cited (PTO-892)
2) ☐ Notice of Draftsperson's Patent Drawing Review (PTO-948)
3) ☐ Information Disclosure Statement(s) (PTO/SB/08) Paper No(s)/Mail Date _____.
4) ☐ Interview Summary (PTO-413) Paper No(s)/Mail Date. _____.
5) ☐ Notice of Informal Patent Application
6) ☐ Other: _____.

Detailed Action

The instant application having Application No. 10/530,375 has a total of 20 claims pending in the application, there are 2 independent claim and 18 dependent claims, all of which are ready for examination by the examiner.

INFORMATION CONCERNING OATH/DECLARATION

Oath/Declaration

1. The applicant's oath/declaration has been reviewed by the examiner and is found to conform to the requirements prescribed in **37 C.F.R. 1.63**.

REJECTIONS NOT BASED ON PRIOR ART

Claim Rejections - 35 USC § 112

2. The following is a quotation of the second paragraph of 35 U.S.C. 112:

> The specification shall conclude with one or more claims particularly pointing out and distinctly claiming the subject matter which the applicant regards as his invention.

3. Claims 11-17 and 20 are rejected under 35 U.S.C. 112, second paragraph, as being indefinite for failing to particularly point out and distinctly claim the subject matter which applicant regards as the invention.

4. Claim 11 recites "updating an offset value in an offset register during the execution of the program by a functional unit of the plurality of functional units." It is unclear whether the functional unit is executing the program or the functional unit is updating an offset value. Claims 12-17 and 20 fail to remedy the deficiency.

REJECTIONS BASED ON PRIOR ART

Claim Rejections - 35 USC § 103

5. The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this Office action:

> (a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negatived by the manner in which the invention was made.

6. Claims 1-5, 7-9, and 11-20 are rejected under 35 U.S.C. 103(a) as being unpatentable over Schlansker'538 (U.S. Patent No. 7,024,538), herein after referred to as Schlansker'538, in view of Fleck et al. (U.S. Patent No. 6,076,159), herein referred to as Fleck et al.'159 and Moroney et al. (U.S. Patent Application Publication No. 2003/0145116), herein referred to as Moroney et al.

7. Referring to claim 1, Schlansker'538 discloses as claimed, a data processing apparatus, the apparatus comprising an instruction addressing unit (**see fig. 4, controller 55, which controls addressing by using lookup table 57, main memory 56, and individual instruction memories**); an instruction memory system (**see fig. 4, regarding main memory 56, and processor specific instruction memories such as**

instruction memory 53) arranged to output an instruction word (**see col. 5, lines 1-4, regarding the main memory holding a super instruction**), capable of containing a plurality of instructions (**see fig. 3, regarding instruction words C1-C5**), in response to an instruction address from the instruction addressing unit (**see col. 5, lines 43-55, regarding the target address being used from the controller to find local instruction memory addresses**), the instruction memory system comprising a plurality of memory units (**see fig. 4, local instruction memory 53, where each processing unit has a local instruction memory**), arranged to output respective parts of the instruction word in parallel (**see col. 5, lines 43-55, regarding the functional units obtaining instructions for processing after a branch, and resuming execution. Also see col. 5, lines 37-42, regarding the processing units stepping through instructions in synchronization**); an instruction execution unit, comprising a plurality of functional units (**see col. 5, lines 27-30, regarding a processor connected to a plurality of processing sections**), each capable of executing a respective instruction from the instruction word in parallel with execution of other instructions from the instruction word by other ones of the functional units (**see col. 3, lines 43-46, regarding sets of instructions from a super instruction each being executed by each processor on each cycle, thus operating in parallel**); an instruction address modification circuit arranged to modify translation of the instruction address into a physical address for a particular one of the memory units relative to other ones of the memory units and to change generation of instruction words from instructions from different memory units during execution of a program (**see col. 6, lines 35-48,**

regarding where the memories are caches, and a tag array is used to translate target addresses to local addresses for each corresponding functional unit for the cache of that functional unit. Also see col. 5, lines 37-42, where the processing units step through the super instruction one row at a time with the row changing).

Schlansker'538 discloses the claimed invention except for the instruction address modification circuit being configured to change generation of instruction words from instructions from different memory units, and to modify an address translation between supplying a first instruction address for a first instruction word and supplying a second instruction address for a second instruction word, the second instruction word being different from the first instruction word and including a copy of a part of the first instruction word, so that the part of the first instruction word is re-used in the second instruction word thereby reducing memory needed to store the program; wherein the instruction address modification circuit includes an offset register connected to an output of a functional unit of the plurality of functional units, the functional unit updating an offset value in the offset register during the execution of the program.

However, Fleck et al.'159 disclose the instruction address modification circuit being configured to modify an address translation between supplying a first instruction address for a first instruction word (**see col. 7, lines 57-65, where the first time a loop is encountered, the loop instruction is fetched and issued**) and supplying a second instruction address for a second instruction word (**see col. 7, lines 66-67 and col. 8, lines 1-9, where a loop cache buffer is set up, and can be executed from the loop**

pipeline), the second instruction word being different from the first instruction word and including a copy of a part of the first instruction word (**adding the loop cache buffer to an execution unit in Schlansker would enable one execution unit to reuse the loop instructions while the remaining part of the VLIW is the same, therefore enabling different instructions with part of a first instruction reused**), so that the part of the first instruction word is re-used in the second instruction word thereby reducing memory needed to store the program (**see col. 8, lines 5-7, where on subsequent iterations, the loop instruction is detected by the loop cache buffer and executed in the loop execution unit. The loop cache buffer allows for reusing instructions**).

Moroney et al. disclose wherein the instruction address modification circuit includes an offset register connected to an output of a functional unit of the plurality of functional units, the functional unit updating an offset value in the offset register during the execution of the program (**see para. 95, where the translate block 74 stores an offset that enables jumping over sections of memory. This offset is sent through the output of the translate block 74 into the controller 119 in fig. 8. This offset would change depending on what jump was needed. Also see para. 89-90**).

Schlansker'538 and Fleck et al.'159 are analogous art because they are from the same field of endeavor of parallel execution in multi-processor systems (**see Schlansker'538, col. 4, lines 30-34, regarding a VLIW processor with multiple functional units. See Fleck et al.'159, col. 1, lines 59-67, regarding a data processor executing with parallel pipelines**).

It would have been obvious to one of ordinary skill in the art at the time the invention was made to modify Schlansker'538 to comprise the instruction address modification circuit being configured to modify an address translation between supplying a first instruction address for a first instruction word and supplying a second instruction address for a second instruction word, the second instruction word being different from the first instruction word and including a copy of a part of the first instruction word, so that the part of the first instruction word is re-used in the second instruction word thereby reducing memory needed to store the program, as taught by Fleck et al.'159, in order to allow for one the execution of a loop and additional code in parallel, and therefore enable highly efficient parallel execution (**see col. 8, lines 8-12, and col. 10, lines 22-26, regarding how the parallel execution of three pipelines is useful for DSP related instructions. Fleck adds the loop pipeline and loop cache buffer for that purpose. It would be obvious to incorporate the loop pipeline and cache buffer into Schlansker's system to allow for faster parallel execution**).

Schlansker'538 and Moroney et al. are analogous art because they are from the same field of endeavor of VLIW systems (**see Schlansker'538, col. 4, lines 30-34, regarding a VLIW processor. See Moroney et al.'116, para. 5, lines 4-12, regarding a system that executes very long instruction words in different functional units in parallel**).

It would have been obvious to one of ordinary skill in the art at the time the invention was made to modify Schlansker'538 to comprise wherein the instruction address modification circuit includes an offset register connected to an output of a

functional unit of the plurality of functional units, the functional unit updating an offset value in the offset register during the execution of the program, as taught by Moroney et al., in order to allow for skipping over selected instructions in the memory (**see para. 95**). Moroney allows for conditional branching, and therefore maintaining state information more effectively (**see para. 46, lines 6-10**).

8. As to claim 2, Schlansker'538 discloses as claimed, the data processing apparatus according to Claim 1, wherein the instruction address modification circuit is arranged to modify the translation under control of a modification update instruction from the instruction word during program execution (**see fig. 3, regarding a super instruction, where the header stipulates how the instruction will be modified and broken down into several different parts or tuples, as stipulated in col. 5, lines 1-25. If there is no tuple for a functional unit, then it is interpreted as a sequence of NOOPs. Therefore, the super instruction header contains information of how the instruction will be modified**).

9. As to claim 3, Schlansker'538 discloses as claimed, the data processing apparatus according to Claim 2, wherein the particular one of the memory units is arranged to supply instructions exclusively to a group that contains a subset of the functional units (**see fig. 4, where functional unit 52 receives instructions from instruction memory 53, and functional unit 52 is a group that is a subset of a number of functional units, as shown by other processor section 51**).

Schlansker'538 discloses the claimed invention except for the group containing a modification update functional unit constructed to execute the modification update instruction.

However, Moroney et al.'116 disclose the group containing a modification update functional unit constructed to execute the modification update instruction (**see para. 76, lines 1-4, regarding an ALU which includes a translator block. Also see para. 82, lines 8-13, regarding the translator block handling jumps based on case statements**).

Schlansker'538 and Moroney et al.'116 are analogous art because they are from the same field of endeavor of VLIW systems (**see Schlansker'538, col. 4, lines 30-34, regarding a VLIW processor. See Moroney et al.'116, para. 5, lines 4-12, regarding a system that executes very long instruction words in different functional units in parallel**).

It would have been obvious to one of ordinary skill in the art at the time the invention was made to modify Schlansker'538 to comprise the group containing a modification update functional unit constructed to execute the modification update instruction, as taught by Moroney et al.'116, in order to enable the functional units to operate in parallel with each other (**see para. 5, lines 6-12, regarding the VLIW system having different functional blocks which enable parallel operation. By having the modification update unit in a functional block, it enables parallel operation, and handles instructions in a similar manner to other functional units**).

10. As to claim 4, Schlansker'538 discloses as claimed, the data processing apparatus according to Claim 2, wherein the particular one of the memory units is arranged to supply instructions exclusively to a group that contains a subset of the functional units (**see fig. 4, where functional unit 52 receives instructions from instruction memory 53, and functional unit 52 is a group that is a subset of a number of functional units, as shown by other processor section 51**), the functional units comprising a modification update functional unit outside the group constructed to execute the modification update instruction (**see fig. 4, regarding the controller 55, which could qualify as a functional unit, and is outside the group of other functional units as shown in fig. 4**).

11. As to claim 5, Schlansker'538 discloses the claimed invention except for a data processing apparatus according to Claim 2 wherein the modification update instruction is a conditional instruction, the modification update being executed dependent on fulfillment of a condition specified in the modification update instruction.

However, Moroney et al.'116 disclose where the modification update instruction is a conditional instruction (**see para. 82, lines 8-13, regarding the translator block handling jumps based on case statements**), the modification update being executed dependent on fulfillment of a condition specified in the modification update instruction (**see para. 83, where a case statement is described. If a comparison is true, then the jump occurs, and the instruction address is changed, otherwise the jump does not occur**).

Schlansker'538 and Moroney et al.'116 are analogous art because they are from the same field of endeavor of VLIW systems **(see Schlansker'538, col. 4, lines 30-34, regarding a VLIW processor. See Moroney et al.'116, para. 5, lines 4-12, regarding a system that executes very long instruction words in different functional units in parallel)**.

It would have been obvious to one of ordinary skill in the art at the time the invention was made to modify Schlansker'538 to comprise where the modification update instruction is a conditional instruction, the modification update being executed dependent on fulfillment of a condition specified in the modification update instruction, as taught by Moroney et al.'116, in order to allow for conditional branching **(see para. 46, lines 6-10, regarding the case statement allowing for conditional branching, and being able to maintain state information more effectively)**.

12. As to claim 7, Schlansker'538 discloses the claimed invention except for a data processing apparatus according to Claim 1, programmed to use repeated modification of said translation to repeatedly output one or more instructions making up a loop of instructions, while the instruction address progresses so that the instructions from the loop are combined in the instruction words with progressive instructions that are not repeated during at least part of the repetitions of supply of instructions from the loop.

However, Fleck et al.'159 discloses a data processing apparatus programmed to use repeated modification of said translation to repeatedly output one or more instructions making up a loop of instructions **(see col. 7, lines 57-67, and col. 8, lines 1-9, regarding a third pipeline which executes a loop. Also see fig. 1, regarding**

loop pipeline 12. More detail is shown in fig. 3, which shows the loop execution unit 4), while the instruction address progresses so that the instructions from the loop are combined in the instruction words with progressive instructions **(see col. 7, lines 1-10, regarding the first and second pipelines which are the data pipeline and the address pipeline. Also see fig. 1, regarding the two major pipelines 10 and 11)** that are not repeated during at least part of the repetitions of supply of instructions from the loop **(see col. 7, lines 60-62, where a loop operation is performed in parallel with an integer and a load/store operation)**.

Schlansker'538 and Fleck et al.'159 are analogous art because they are from the same field of endeavor of parallel execution in multi-processor systems **(see Schlansker'538, col. 4, lines 30-34, regarding a VLIW processor with multiple functional units. See Fleck et al.'159, col. 1, lines 59-67, regarding a data processor executing with parallel pipelines)**.

It would have been obvious to one of ordinary skill in the art at the time the invention was made to modify Schlansker'538 to comprise where the modification update instruction is a conditional instruction, the modification update being executed dependent on fulfillment of a condition specified in the modification update instruction, as taught by Fleck et al.'159, in order to allow for one the execution of a loop and additional code in parallel, and therefore enable highly efficient parallel execution **(see col. 8, lines 8-12, and col. 10, lines 22-26, regarding how the parallel execution of three pipelines is useful for DSP related instructions)**.

13. As to claim 8, Schlansker'538 discloses as claimed, the data processing apparatus according to Claim 2, programmed to use said modification update instruction (**see col. 5, lines 1-25, where an instruction word is broken into tuples**) to selectively output, dependent on a data dependent condition (**see col. 5, lines 1-25, where depending on if the functional unit is idle, data might be sent. Otherwise NOOPs are given.**), a first or a second block of one or more instructions from said particular one of the memory units (**see fig. 4, lookup table 57, where an address for a functional unit 1 would link and point to instructions. See fig. 3, where blocks of instructions are shown in the instruction word**), while the instruction address progresses so that at least part of the memory units output one or more instructions from a third block of instructions (**see fig. 4, lookup table 57, where an address for a functional unit n would link to different instructions than the first. Also see fig. 3, where C5 represents different instructions from C1**) as part of the instruction word or words in combination with instructions from said first or second block (**see col. 5, lines 1-25, where each of these parts of instructions are tuples, and part of one instruction word**).

14. As to claim 9, Schlansker'538 discloses as claimed, the data processing apparatus according to Claim 1, wherein a first number of addressable instruction addresses of the particular one of the memory units differs from a second number of addressable instruction addresses of at least one of the memory units (**see fig. 4, where lookup table 57 gives separate addresses for each functional unit based on a common block address**).

15. As to claim 11, Schlansker'538 discloses as claimed, a method of executing a program of instruction words with a data processing apparatus that comprises a plurality of functional units (**see col. 5, lines 27-30, regarding a processor connected to a plurality of processing sections**) capable of executing a plurality of instructions from each instruction word in parallel (**see col. 3, lines 43-46, regarding sets of instructions from a super instruction each being executed by each processor on each cycle, thus operating in parallel**), wherein the instructions from each of at least some of the instruction words are fetched from respective memory units in parallel (**see col. 7, lines 32-36, regarding parallel lookups for operations for each functional unit, and then each unit begins interpreting the stream of operations**), the method comprising the acts of: addressing the instruction word with an instruction address that is common for the functional units (**see col. 3, lines 42-47, regarding broadcasting the address of the super instruction to all functional units during a branch**); using a modifiable translation of the instruction address into a physical address for a particular one of the memory units (**see col. 6, lines 35-48, regarding where the memories are caches, and a tag array is used to translate target addresses to local addresses for each corresponding functional unit for the cache of that functional unit**) to select dependent on program execution which instructions from the memory units will be combined into the instruction word in response to the instruction address (**see col. 5, lines 1-25, regarding tuples from the super instruction going to individual memory units. If a functional unit remains idle, which could be dependent on program execution, then NOOPs would be used as a sequence of operations for a**

particular functional unit. Otherwise, the tuples from the super instruction can be used).

Schlansker'538 discloses the claimed invention except for modifying an address translation between supplying a first instruction address for a first instruction word and supplying a second instruction address for a second instruction word, the second instruction word being different from the first instruction word and including a copy of a part of the first instruction word, so that a part of the first instruction word is re-used in the second instruction word thereby reducing memory needed to store the program, wherein the modifying acts includes updating an offset value in an offset register during the execution of the program by a functional unit of the plurality of functional units, the offset register being connected to an output of the functional.

However, Fleck et al.'159 disclose modifying an address translation between supplying a first instruction address for a first instruction word (**see col. 7, lines 57-65, where the first time a loop is encountered, the loop instruction is fetched and issued**) and supplying a second instruction address for a second instruction word (**see col. 7, lines 66-67 and col. 8, lines 1-9, where a loop cache buffer is set up, and can be executed from the loop pipeline**), the second instruction word being different from the first instruction word and including a copy of a part of the first instruction word (**adding the loop cache buffer to an execution unit in Schlansker would enable one execution unit to reuse the loop instructions while the remaining part of the VLIW is the same, therefore enabling different instructions with part of a first instruction reused**), so that a part of the first instruction word is re-used in the second

instruction word thereby reducing memory needed to store the program (**see col. 8, lines 5-7, where on subsequent iterations, the loop instruction is detected by the loop cache buffer and executed in the loop execution unit. The loop cache buffer allows for reusing instructions**).

Moroney et al. disclose wherein the modifying acts includes updating an offset value in an offset register during the execution of the program by a functional unit of the plurality of functional units, the offset register being connected to an output of the functional (**see para. 95, where the translate block 74 stores an offset that enables jumping over sections of memory. This offset is sent through the output of the translate block 74 into the controller 119 in fig. 8. This offset would change depending on what jump was needed. Also see para. 89-90**).

Schlansker'538 and Fleck et al.'159 are analogous art because they are from the same field of endeavor of parallel execution in multi-processor systems (**see Schlansker'538, col. 4, lines 30-34, regarding a VLIW processor with multiple functional units. See Fleck et al.'159, col. 1, lines 59-67, regarding a data processor executing with parallel pipelines**).

It would have been obvious to one of ordinary skill in the art at the time the invention was made to modify Schlansker'538 to modifying an address translation between supplying a first instruction address for a first instruction word and supplying a second instruction address for a second instruction word, the second instruction word being different from the first instruction word and including a copy of a part of the first instruction word, so that a part of the first instruction word is re-used in the second

instruction word thereby reducing memory needed to store the program, as taught by Fleck et al.'159, in order to allow for one the execution of a loop and additional code in parallel, and therefore enable highly efficient parallel execution **(see col. 8, lines 8-12, and col. 10, lines 22-26, regarding how the parallel execution of three pipelines is useful for DSP related instructions. Fleck adds the loop pipeline and loop cache buffer for that purpose. It would be obvious to incorporate the loop pipeline and cache buffer into Schlansker's system to allow for faster parallel execution)**.

Schlansker'538 and Moroney et al. are analogous art because they are from the same field of endeavor of VLIW systems **(see Schlansker'538, col. 4, lines 30-34, regarding a VLIW processor. See Moroney et al.'116, para. 5, lines 4-12, regarding a system that executes very long instruction words in different functional units in parallel)**.

It would have been obvious to one of ordinary skill in the art at the time the invention was made to modify Schlansker'538 to comprise wherein the modifying acts includes updating an offset value in an offset register during the execution of the program by a functional unit of the plurality of functional units, the offset register being connected to an output of the functional, as taught by Moroney et al., in order to allow for skipping over selected instructions in the memory **(see para. 95)**. Moroney allows for conditional branching, and therefore maintaining state information more effectively **(see para. 46, lines 6-10)**.

16. As to claim 12, Schlansker'538 discloses as claimed, the method of executing a program of instruction words according to Claim 11, wherein the modifiable translation

is selected under control of a modification update instruction in the program (**see fig. 3, regarding a super instruction, where the header stipulates how the instruction will be modified and broken down into several different parts or tuples, as stipulated in col. 5, lines 1-25. If there is no tuple for a functional unit, then it is interpreted as a sequence of NOOPs. Therefore, the super instruction header contains information of how the instruction will be modified**).

17. As to claim 13, Schlansker'538 discloses the claimed invention except for a method of executing a program of instruction words according to Claim 11, comprising modifiable translation to repeatedly fetch instructions from a loop from repeated physical addresses in a particular one of the memory units in response to progressive instruction addresses, so that the instructions from at least part of repetitions of the loop are combined in the instruction words with progressively different instructions memory units other the than the particular one of the memory unit.

However, Fleck et al.'159 discloses a method of executing a program of instruction words according to Claim 11, comprising modifiable translation to repeatedly fetch instructions from a loop from repeated physical addresses (**see col. 7, lines 64-65, and col. 8, lines 5-7, regarding an instruction being fetched the first time, but loaded into a buffer for subsequent loop operations, which would mean in subsequent loop operations, the address is being fetched from a repeated address**) in a particular one of the memory units in response to progressive instruction addresses (**see col. 2, lines 8-29, regarding the steps of executing an instruction, where the instructions would be executed progressively**), so that the instructions

from at least part of repetitions of the loop are combined in the instruction words with progressively different instructions memory units other the than the particular one of the memory unit (**see col. 7, lines 60-62, where a loop operation is performed in parallel with an integer and a load/store operation**).

Schlansker'538 and Fleck et al.'159 are analogous art because they are from the same field of endeavor of parallel execution in multi-processor systems (**see Schlansker'538, col. 4, lines 30-34, regarding a VLIW processor with multiple functional units. See Fleck et al.'159, col. 1, lines 59-67, regarding a data processor executing with parallel pipelines**).

It would have been obvious to one of ordinary skill in the art at the time the invention was made to modify Schlansker'538 to comprise a method of executing a program of instruction words according to Claim 11, comprising modifiable translation to repeatedly fetch instructions from a loop from repeated physical addresses in a particular one of the memory units in response to progressive instruction addresses, so that the instructions from at least part of repetitions of the loop are combined in the instruction words with progressively different instructions memory units other the than the particular one of the memory unit, as taught by Fleck et al.'159, in order to allow for one the execution of a loop and additional code in parallel, and therefore enable highly efficient parallel execution (**see col. 8, lines 8-12, and col. 10, lines 22-26, regarding how the parallel execution of three pipelines is useful for DSP related instructions**).

18. As to claim 14, Schlansker'538 discloses as claimed, the computer program product comprising instruction words **(see col. 5, lines 1-2, regarding the main memory containing super instructions)**, each for execution in one or more respective instruction cycles by a data processing apparatus **(see col. 5, lines 39-41, regarding the processing sections stepping through the instructions on each cycle)** according to Claim 2, the instruction words comprising at least one modification update instruction **(see fig. 3, regarding a super instruction, where the header stipulates how the instruction will be modified and broken down into several different parts or tuples, as stipulated in col. 5, lines 1-25. If there is no tuple for a functional unit, then it is interpreted as a sequence of NOOPs. Therefore, the super instruction header contains information of how the instruction will be modified)** for causing the data processing apparatus to execute, upon addressing a first one of the instruction words, a combination of instructions from the first one of the instruction word **(see fig. 3, where the instructions to functional unit 1, or C1 would constitute first instruction words)** with further instructions from outside the first instruction word **(see fig. 3, where instructions to functional unit 2, or C2 would constitute instructions from outside the first instruction word)**, following execution of the modification update word **(see col. 5, lines 1-25, regarding first interpreting the super instruction block, including separating tuples, and inserting NOOPs into idle functional units)**.

19. As to claim 15, Schlansker'538 discloses the claimed invention except for the data processing apparatus of claim 1, wherein the instruction address modification

circuit is configured to modify the translation in response to an adjust output from one of the plurality of the functional units.

However, Fleck et al.'159 disclose wherein the instruction address modification circuit is configured to modify the translation in response to an adjust output from one of the plurality of the functional units (**see col. 7, lines 64-67 and col. 8, lines 1-9, where a loop instruction is detected, and then executed in the loop execution unit of the loop pipeline. Detecting a loop instruction would therefore cause an adjustment, as the loop instruction would be executed in the loop pipeline**).

Schlansker'538 and Fleck et al.'159 are analogous art because they are from the same field of endeavor of parallel execution in multi-processor systems (**see Schlansker'538, col. 4, lines 30-34, regarding a VLIW processor with multiple functional units. See Fleck et al.'159, col. 1, lines 59-67, regarding a data processor executing with parallel pipelines**).

It would have been obvious to one of ordinary skill in the art at the time the invention was made to modify Schlansker'538 to comprise wherein the instruction address modification circuit is configured to modify the translation in response to an adjust output from one of the plurality of the functional units, as taught by Fleck et al.'159, in order to allow for one the execution of a loop and additional code in parallel, and therefore enable highly efficient parallel execution (**see col. 8, lines 8-12, and col. 10, lines 22-26, regarding how the parallel execution of three pipelines is useful for DSP related instructions**).

20. As to claim 16, Schlansker'538 discloses the claimed invention except for the data processing apparatus of claim 1, wherein the instruction address modification circuit is operationally coupled to a controller that provides the instruction address, and to one of the plurality of the functional units that provides an adjust signal to the instruction address modification circuit; the instruction address modification circuit being configured to modify the translation in response to the adjust output and to provide a modified translated address to one of the plurality of the memory units.

However, Fleck et al.'159 disclose wherein the instruction address modification circuit is operationally coupled to a controller that provides the instruction address (**see fig. 3, where the address modification circuit, which could be represented by the loop cache/loop execution is coupled to pc update and control**), and to one of the plurality of the functional units that provides an adjust signal to the instruction address modification circuit (**see fig. 3, showing that the functional units, including the loop execution unit are coupled to the loop cache, pc update and control, the pre-decoder and instruction demux**); the instruction address modification circuit being configured to modify the translation in response to the adjust output and to provide a modified translated address to one of the plurality of the memory units (**see col. 7, lines 64-67 and col. 8, lines 1-9, where a loop instruction is detected, and then executed in the loop execution unit of the loop pipeline. Detecting a loop instruction would therefore cause an adjustment, as the loop instruction would be executed in the loop pipeline**).

Schlansker'538 and Fleck et al.'159 are analogous art because they are from the same field of endeavor of parallel execution in multi-processor systems (**see Schlansker'538, col. 4, lines 30-34, regarding a VLIW processor with multiple functional units. See Fleck et al.'159, col. 1, lines 59-67, regarding a data processor executing with parallel pipelines**).

It would have been obvious to one of ordinary skill in the art at the time the invention was made to modify Schlansker'538 to comprise wherein the instruction address modification circuit is operationally coupled to a controller that provides the instruction address, and to one of the plurality of the functional units that provides an adjust signal to the instruction address modification circuit; the instruction address modification circuit being configured to modify the translation in response to the adjust output and to provide a modified translated address to one of the plurality of the memory units, as taught by Fleck et al.'159, in order to allow for one the execution of a loop and additional code in parallel, and therefore enable highly efficient parallel execution (**see col. 8, lines 8-12, and col. 10, lines 22-26, regarding how the parallel execution of three pipelines is useful for DSP related instructions**).

21. As to claim 17, Schlansker'538 discloses the claimed invention except for the method of claim 11, wherein the modifying act is performed in response to an adjust output from one of the plurality of the functional units.

However, Fleck et al.'159 disclose wherein the modifying act is performed in response to an adjust output from one of the plurality of the functional units (**see col. 7, lines 64-67 and col. 8, lines 1-9, where a loop instruction is detected, and then**

executed in the loop execution unit of the loop pipeline. Detecting a loop instruction would therefore cause an adjustment, as the loop instruction would be executed in the loop pipeline).

Schlansker'538 and Fleck et al.'159 are analogous art because they are from the same field of endeavor of parallel execution in multi-processor systems (**see Schlansker'538, col. 4, lines 30-34, regarding a VLIW processor with multiple functional units. See Fleck et al.'159, col. 1, lines 59-67, regarding a data processor executing with parallel pipelines**).

It would have been obvious to one of ordinary skill in the art at the time the invention was made to modify Schlansker'538 to comprise wherein the modifying act is performed in response to an adjust output from one of the plurality of the functional units, as taught by Fleck et al.'159, in order to allow for one the execution of a loop and additional code in parallel, and therefore enable highly efficient parallel execution (**see col. 8, lines 8-12, and col. 10, lines 22-26, regarding how the parallel execution of three pipelines is useful for DSP related instructions**).

22. As to claim 18, Schlansker'538 discloses the claimed invention except for the data processing apparatus of claim 1, wherein an output of the offset register is connected to an offset adder, the offset adder being connected between a controller that provides the instruction address and the particular one of the memory units.

However Moroney et al. disclose wherein an output of the offset register is connected to an offset adder, the offset adder being connected between a controller that provides the instruction address and the particular one of the memory units (**see**

para. 95, where the offset is sent to the controller and therefore allows for skipping over portions of memory. An adder would need to be used in order to combine the offset with the address. Each functional unit in Schlansker'538 has a corresponding memory unit. Therefore the adder would be between the memory unit and the controller).

Schlansker'538 and Moroney et al.'116 are analogous art because they are from the same field of endeavor of VLIW systems **(see Schlansker'538, col. 4, lines 30-34, regarding a VLIW processor. See Moroney et al.'116, para. 5, lines 4-12, regarding a system that executes very long instruction words in different functional units in parallel).**

It would have been obvious to one of ordinary skill in the art at the time the invention was made to modify Schlansker'538 to comprise wherein an output of the offset register is connected to an offset adder, the offset adder being connected between a controller that provides the instruction address and the particular one of the memory units, as taught by Moroney et al.'116, in order to add the value of the offset to the address, enabling branching and jumping **(see para. 46, lines 6-10, regarding the case statement allowing for conditional branching, and being able to maintain state information more effectively).**

23. As to claim 19, Schlansker'538 discloses the claimed invention except for the data processing apparatus of claim 1, wherein the functional unit updates the offset value during the execution of the program dependent on conditions that occur during execution.

However Moroney et al. disclose wherein the functional unit updates the offset value during the execution of the program dependent on conditions that occur during execution (**see para. 89 regarding jumps and branches that occur during program execution. See para. 46, regarding conditional branching**).

Schlansker'538 and Moroney et al.'116 are analogous art because they are from the same field of endeavor of VLIW systems (**see Schlansker'538, col. 4, lines 30-34, regarding a VLIW processor. See Moroney et al.'116, para. 5, lines 4-12, regarding a system that executes very long instruction words in different functional units in parallel**).

It would have been obvious to one of ordinary skill in the art at the time the invention was made to modify Schlansker'538 to comprise wherein the functional unit updates the offset value during the execution of the program dependent on conditions that occur during execution, as taught by Moroney et al.'116, in order to enable branching and jumping (**see para. 46, lines 6-10, regarding the case statement allowing for conditional branching, and being able to maintain state information more effectively**).

24. As to claim 20, Schlansker'538 discloses the claimed invention except for the method of claim 11, further comprising connecting an output of the offset register to an offset adder, the offset adder being connected between a controller that provides the instruction address and the particular one of the memory units.

However Moroney et al. disclose comprising connecting an output of the offset register to an offset adder, the offset adder being connected between a controller that

provides the instruction address and the particular one of the memory units (**see para. 95, where the offset is sent to the controller and therefore allows for skipping over portions of memory. An adder would need to be used in order to combine the offset with the address. Each functional unit in Schlansker'538 has a corresponding memory unit. Therefore the adder would be between the memory unit and the controller**).

Schlansker'538 and Moroney et al.'116 are analogous art because they are from the same field of endeavor of VLIW systems (**see Schlansker'538, col. 4, lines 30-34, regarding a VLIW processor. See Moroney et al.'116, para. 5, lines 4-12, regarding a system that executes very long instruction words in different functional units in parallel**).

It would have been obvious to one of ordinary skill in the art at the time the invention was made to modify Schlansker'538 to comprise comprising connecting an output of the offset register to an offset adder, the offset adder being connected between a controller that provides the instruction address and the particular one of the memory units, as taught by Moroney et al.'116, in order to add the value of the offset to the address, enabling branching and jumping (**see para. 46, lines 6-10, regarding the case statement allowing for conditional branching, and being able to maintain state information more effectively**).

25. Claim 6 is rejected under 35 U.S.C. 103(a) as being unpatentable over Schlansker'538, in view of Fleck et al.'159 and Moroney et al., as applied to claim 1, and further in view of the examiner taking official notice.

26. As to claim 6, Schlansker'538 discloses as claimed, a data processing apparatus according to Claim 1, wherein the instruction address modification circuit is arranged for instruction address and memory unit dependent address translation (**see fig. 4, lookup table 57, which shows that the translation is dependent on a memory unit corresponding to a functional unit, and is found by giving a block address**), so that a first and a second instruction address are to mutually different physical addresses for one or more memory units other than the particular one of the memory traits (**see fig. 4, lookup table 57, where a given block address is given, and addresses corresponding to functional units are found. Also see col. 5, lines 47-55**).

Schlansker'538 discloses the claimed invention except where a first and a second instruction address are translated to a same physical address for the particular one of the memory units.

However, it is well known in the art that virtual address to physical address translation enables two different virtual addresses to share the same physical address when translated. As an example, Tremblay (U.S. Patent No. 5,875,483), herein referred to as Tremblay, teaches that multiple virtual addresses can point to the same physical address when translated (**see col. 3, lines 54-63, discussing converting virtual addresses to physical addresses)**.

It would have been obvious to one of ordinary skill in the art at the time the invention was made to modify Schlansker'538 to comprise where a first and a second instruction address are translated to a same physical address for the particular one of the memory units, in order to use virtual addressing, and therefore map non-contiguous

memory through the use of contiguous virtual addresses, or to allow for easier mapping across multiple memories through virtual addressing.

27. Claim 10 is rejected under 35 U.S.C. 103(a) as being unpatentable over Schlansker'538, in view of Fleck et al.'159 and Moroney et al., as applied to claim 1, and further in view of Matsushima (U.S. Patent No. 5,208,781), herein referred to as Matsushima'781.

28. As to claim 10, Schlansker'538 discloses the claimed invention except the data processing apparatus according to Claim 9, wherein the particular one of the memory units is arranged to switch to a power saving state when the modified instruction address is outside an address range or set of address ranges that contains said first number of instruction addresses.

However, Matsushima'781 discloses where the particular one of the memory units is arranged to switch to a power saving state when the modified instruction address is outside an address range or set of address ranges that contains said first number of instruction addresses **(see col. 3, lines 15-26, regarding switching to a power saving state if the address decoder outputs an address outside the range of addresses of a memory chip)**.

Schlansker'538 and Matsushima'781 are analogous art because they are both attempting to solve the problem of using memory more efficiently in a system (**see Schlansker'538, col. 2, lines 11-13, regarding the problem of using too much memory in standard VLIW programs. See Matsushima'781, col. 3, lines 15-26, regarding conserving power for memory chips**).

It would have been obvious to one of ordinary skill in the art at the time the invention was made to modify Schlansker'538 to comprise where the particular one of the memory units is arranged to switch to a power saving state when the modified instruction address is outside an address range or set of address ranges that contains said first number of instruction addresses, as taught by Matsushima'781, in order to conserve power when the memory unit isn't being used (**see col. 3, lines 15-26, regarding the purpose being to save power which would be consumed by the memory chip**).

ACKNOWLEDGEMENT OF ISSUES RAISED BY THE APPLICANT

Response to Amendment

29. Applicant's arguments filed 12/8/2008 have been fully considered but they are not persuasive.

30. Applicant argues with respect to claim 1 that Schlansker and Fleck do not teach an offset register connected to the output of the functional unit. However, Moroney teaches that a functional unit for conditional branching can store an offset. Therefore, its offset register is connected to the functional unit and updated throughout the execution of the program for conditional branches and jumps.

CLOSING COMMENTS

Conclusion

a. STATUS OF CLAIMS IN THE APPLICATION

31. The following is a summary of the treatment and status of all claims in the application as recommended by **M.P.E.P. 707.07(i)**:

a(1) CLAIMS REJECTED IN THE APPLICATION

32. Per the instant office action, claims 1-20 have received a second action on the merits and are the subject of a final rejection.

33. **THIS ACTION IS MADE FINAL.** Applicant is reminded of the extension of time policy as set forth in 37 CFR 1.136(a).

A shortened statutory period for reply to this final action is set to expire THREE MONTHS from the mailing date of this action. In the event a first reply is filed within TWO MONTHS of the mailing date of this final action and the advisory action is not mailed until after the end of the THREE-MONTH shortened statutory period, then the shortened statutory period will expire on the date the advisory action is mailed, and any extension fee pursuant to 37 CFR 1.136(a) will be calculated from the mailing date of the advisory action. In no event, however, will the statutory period for reply expire later than SIX MONTHS from the mailing date of this final action.

b. DIRECTION OF FUTURE CORRESPONDENCES

34. Any inquiry concerning this communication or earlier communications from the examiner should be directed to ALAN M. OTTO whose telephone number is 571-270-1626. The examiner can normally be reached on 8:00-5:30 M-F.

If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, Kevin Ellis can be reached on 571-272-4205. The fax phone number for the organization where this application or proceeding is assigned is 571-273-8300.

Information regarding the status of an application may be obtained from the Patent Application Information Retrieval (PAIR) system. Status information for published applications may be obtained from either Private PAIR or Public PAIR. Status information for unpublished applications is available through Private PAIR only. For more information about the PAIR system, see http://pair-direct.uspto.gov. Should you have questions on access to the Private PAIR system, contact the Electronic Business Center (EBC) at 866-217-9197 (toll-free). If you would like assistance from a USPTO Customer Service Representative or access to the automated information system, call 800-786-9199 (IN USA OR CANADA) or 571-272-1000.

***
/Alan M Otto/
Examiner, Art Unit 2187

/Kevin L Ellis/
Acting SPE of Art Unit 2187